Advances in Parametric Real-Time Reasoning[⋆]

Daniel Bundala and Joël Ouaknine

Department of Computer Science, University of Oxford
Wolfson Building, Parks Road, Oxford, OX1 3QD, UK

Abstract. We study the decidability and complexity of the reachability problem in parametric timed automata. The problem was introduced 20 years ago by Alur, Henzinger, and Vardi in [1], where they showed decidability in the case of a single parametric clock, and undecidability for timed automata with three or more parametric clocks.

By translating such problems as reachability questions in certain extensions of parametric one-counter machines, we show that, in the case of two parametric clocks (and arbitrarily many nonparametric clocks), reachability is decidable for parametric timed automata with a single parameter, and is moreover $\mathsf{PSPACE}^{\mathsf{NEXP}}$-hard. In addition, in the case of a single parametric clock (with arbitrarily many nonparametric clocks and arbitrarily many parameters), we show that the reachability problem is NEXP-complete, improving the nonelementary decision procedure of Alur *et al.*

1 Introduction

The problem of reachability in parametric timed automata (PTA) was introduced over two decades ago in a seminal paper of Alur, Henzinger, and Vardi [1]: given a timed automaton in which some of the constants appearing within guards on transitions are parameters, is there some assignment of integers to the parameters such that an accepting location of the resulting concrete timed automaton becomes reachable?

In this framework, a clock is said to be *nonparametric* if it is never compared with a parameter, and *parametric* otherwise. Alur *et al.* [1] showed that, for timed automata with a single parametric clock, reachability is decidable (irrespective of the number of nonparametric clocks). The decision procedure given in [1] however has provably nonelementary complexity. In addition, [1] showed that reachability becomes undecidable for timed automata with at least three parametric clocks.

The decidability of reachability for PTAs with *two* parametric clocks (and arbitrarily many nonparametric clocks) was left open in [1], with hardly any progress (partial or otherwise) that we are aware of in the intervening period. The problem was shown in [1] to subsume the question of reachability in Ibarra *et al.*'s "simple programs" [9], also open for over 20 years, as well as a decision problem for a fragment of Presburger arithmetic with divisibility.

[⋆] Full version of the paper is available at: `http://www.cs.ox.ac.uk/people/joel.ouaknine/publications/advances_parametric14abs.html`

Our main results are as follows: (i) We show that, in the case of two parametric clocks (and arbitrarily many nonparametric clocks), reachability is decidable for PTAs with a *single* parameter. Furthermore, we establish a $\mathsf{PSPACE}^{\mathsf{NEXP}}$ lower bound on the complexity of this problem. (ii) In the case of a single parametric clock (with arbitrarily many nonparametric clocks and arbitrarily many parameters), we show that the reachability problem is NEXP-complete, improving the nonelementary decision procedure of Alur *et al.*

Our results rest in part on new developments in the theory of one-counter machines [5], their encodings in Presburger arithmetic [4], and their application to reachability in (ordinary) timed automata [6,3]. We achieve this by restricting our attention to PTAs with *closed* (i.e. *non-strict*) clock constraints. As parameters are restricted to ranging over integers[1], standard digitisation techniques apply [7,15], reducing the reachability problem over dense time to discrete (integer) time. (Alternatively, our results also apply directly to timed automata interpreted over discrete time, regardless of the type of constraints used.) The restriction to integer time enables us, among others, to keep track of the values of two parametric clocks using a single counter, in effect reducing the reachability problem for timed automata with two parametric clocks to a halting problem for parametric one-counter machines.

Related Work. The decidability of reachability for PTAs can be achieved in certain restricted settings, for instance by bounding the allowed range of the parameters [10] or by requiring that parameters only ever appear either as upper or lower bounds, but never as both [8]: in the latter case, if there is a solution at all then there is one in which parameters are set either to zero or infinity. The primary concern in such restricted settings is usually the development of practical verification tools, and indeed the resulting algorithms tend to have comparatively good complexity.

Miller [14] observed that over dense time and with parameters allowed to range over rational numbers, reachability for PTA becomes undecidable already with a single parametric clock. In the same setting, Doyen [2] showed undecidability of reachability for two parametric clocks even when using exclusively open (i.e. strict) time constraints.

A connection between timed automata and counter machines was previously established in nonparametric settings [6], and used to show that reachability for (ordinary) two-clock timed automata is polynomial-time equivalent to the halting problem for one-counter machines, even when constants are encoded in binary. Unfortunately, it is not obvious how to extend and generalise this construction to PTA, specifically in the case of two parametric clocks and an arbitrary number of nonparametric clocks, as we handle in the present paper. The reduction of [6] was used in [3] to show that halting for bounded one-counter machines, and

[1] Other researchers have considered variations in which parameters are allowed to range over rationals, yielding different outcomes as regards the decidability of reachability; see, e.g., [14,2], discussed further below.

hence reachability for two-clock timed automata, is PSPACE-complete, solving what had been a longstanding open problem.

Finally, parametric one-counter machines without upper bounds imposed on the value of the counter were studied in [5], where reachability was shown to be decidable. The techniques used in [5] make crucial use of the unboundedness of the counter and therefore do not appear applicable in the present setting.

2 Preliminaries

We now give definitions used throughout the rest of the paper. A ***timed automaton*** is a finite automaton extended with clocks; each clock measuring the time since it was last reset. A parametric timed automaton is obtained by replacing the known constants in the guards by parameters.

Formally, let P be a finite set of ***parameters***. An ***assignment*** for P is a function $\gamma : P \to \mathbb{N}$ assigning a *natural number* to each parameter. A ***parametric timed automaton (PTA)*** $A = (S, s_0, C, P, F, E)$ is a tuple where S is the set of states, $s_0 \in S$ is the initial state, C is the set of clocks, P is the set of parameters, $F \subseteq S$ is the set of final states and $E \subseteq S \times S \times 2^C \times G(C, P)$ is the set of edges where $G(C, P)$ is the set of guards of the form $x \leq v, x \geq v$ where x is a clock and $v \in \mathbb{N} \cup P$. An edge (s, s', R, G) is from state s to state s'. Set R specifies which clocks are reset. A clock is ***parametrically constrained*** if it is compared to a parameter in some guard. The class of PTAs with k parametrically constrained clocks is denoted k-PTA. If γ is an assignment to parameters then A^γ denotes the automaton obtained by setting each parameter $p \in P$ to $\gamma(p)$.

A ***configuration*** (s, ν) of A^γ consists of state s and function $\nu : C \to \mathbb{N}$ assigning a value to each clock. A transition exists from configuration (s, ν) to (s', ν') in A^γ, written $(s, \nu) \to (s', \nu')$, if either there is $t \in \mathbb{N}$ such that $\nu(c) + t = \nu'(c)$ for every clock $c \in C$ or there is an edge $e = (s, s', R, G) \in E$ such that G is satisfied for current clock values and if $c \in R$ then $\nu'(c) = 0$ and if $c \notin R$ then $\nu'(c) = \nu(c)$.

The initial clock valuation ν_0 assigns 0 to every clock. A ***run*** of a machine is a sequence $\pi = \boldsymbol{c_1}, \boldsymbol{c_2}, \ldots, \boldsymbol{c_k}$ of configurations such that $\boldsymbol{c_i} \to \boldsymbol{c_{i+1}}$ for each i. A run is called ***accepting*** if $\boldsymbol{c_1}$ is the initial configuration (s_0, ν_0) and $\boldsymbol{c_k}$ is in a final state. The ***existential halting problem***, also known as parametric reachability or the emptiness problem, asks whether there is some parameter valuation γ such that A^γ has an accepting run. From here onwards, we omit "existential" and write simply "halting problem". We say that two automata A_1 and A_2 have ***equivalent halting problem*** if A_1 halts if and only if A_2 halts.

Given a run π, we use $start(\pi) = \boldsymbol{c_1}$ and $end(\pi) = \boldsymbol{c_k}$ to denote the first and the last configuration of the run, respectively. If τ is a run, we write $\pi \to \tau$ if the runs can be connected by a transition, i.e. $end(\pi) \to start(\tau)$.

A ***parametric timed 0/1 automaton*** [1] $A = (S, s_0, C, P, F, E)$ is a timed automaton such that each edge $e \in E$ is labeled by a time increment $t \in \{0, 1\}$. A transition from (s, ν) to (s', ν') is valid only if $\nu'(c) - \nu(c) = t$ for each $c \in C$ not reset by the edge giving rise to the transition.

A ***one-counter machine*** is a finite-state machine equipped with a single counter. Each edge is labelled by an integer, which is added to the counter whenever that edge is taken. The counter is required to be nonnegative at all times. E.g., subtracting $c \in \mathbb{N}$ in one transition and adding c in the next transition leaves the counter unchanged but can be performed only if the counter is at least c.

A ***bounded one-counter machine*** also allows $\leq x$ edges. Such an edge can be taken only when the counter is at most x. Reachability in these two classes of counter machines are respectively known to be NP-complete [5] and PSPACE-complete [3] if the numbers are encoded in binary.

Parametric machines are obtained by replacing the known constants by parameters. A ***parametric bounded one-counter machine (PBOCA)*** $C = (S, s_0, F, P, E, \lambda)$ is a tuple where S is the set of states, s_0 is the initial state, $F \subseteq S$ are the final states, P is the set of parameters, $E \subseteq S \times S$ is the set of edges and $\lambda : E \to Op$ assigns an operation to each edge and has codomain Op: $\{+c, -c, +p, -p, \leq c, = c, \geq c, \leq p, = p, \geq p, +[0,p], \equiv 0 \bmod c \ : \ c \in \mathbb{N}, p \in P\}$.

A ***parametric one-counter machine*** allows only operations: $\pm c, \pm p, \geq c, \geq p, = 0$. Note that parametric one-counter machines are a subclass of parametric bounded one-counter machines.

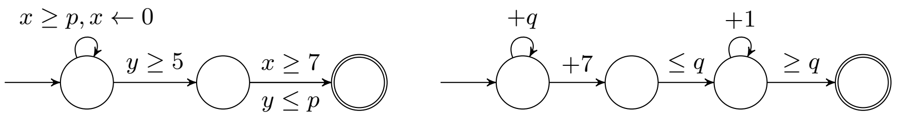

Fig. 1. A parametric timed automaton (left) and a parametric bounded one-counter machine (right). The final states are reachable if, for example, $p = 10$ and $q = 11$.

A ***configuration*** (s, x) of C consists of a state $s \in S$ and counter value $x \in \mathbb{N}$. Thus, the counter is always nonnegative. Machine C starts in state s_0 and counter equal to 0 and then takes individual edges updating the counter. We use $\mathit{counter}(s, x) = x$ to denote the counter value in a configuration. We extend the definition to runs componentwise and write $\mathit{counter}(\pi) \leq C$ (resp. $\mathit{counter}(\pi) \geq C$) if the comparison holds for every element: $\forall i \, . \, \mathit{counter}(\pi(i)) \leq C$ (resp. $\forall i \, . \, \mathit{counter}(\pi(i)) \geq C$).

Let Z be a (nonparametric) one-counter machine. For configurations $\boldsymbol{c}, \boldsymbol{d}$ of Z and numbers $x, y \in \mathbb{N}$, we write $(\boldsymbol{c}, \boldsymbol{d}) \in Z(x, y)$ if there is a run $\pi : \boldsymbol{c} \to \boldsymbol{d}$ such that the counter stays between x and y, i.e. $x < \mathit{counter}(\pi) < y$.

For a parameter assignment γ, configuration (s', x') is directly reachable from (s, x) (written $(s, x) \to (s', x')$) in C^γ if an edge $e = (s, s') \in E$ exists such that

- if $\lambda(e) = \pm c, c \in \mathbb{N}$ then $x \pm c = x'$
- if $\lambda(e) = \pm p, p \in P$ then $x \pm \gamma(p) = x'$
- if $\lambda(e) = \sim c, c \in \mathbb{N}$ then $x = x'$ and $x \sim c$ where $\sim \in \{\leq, \geq\}$
- if $\lambda(e) = \sim p, p \in P$ then $x = x'$ and $x \sim \gamma(p)$ where $\sim \in \{\leq, \geq\}$
- if $\lambda(e) = +[0, p], p \in P$ then $x \leq x' \leq x + \gamma(p)$
- if $\lambda(e) = \equiv 0 \bmod c, x \in \mathbb{Z}$ then $x = x'$ and $x \equiv 0 \bmod c$

The ***existential halting problem*** asks whether there is a parameter valuation γ such that C^γ has an accepting run.

2.1 Presburger Arithmetic

Presburger Arithmetic with Divisibility is the first-order logical theory of $\langle \mathbb{N}, <, +, |, 0, 1\rangle$. The existential fragment (formulae of the form $\exists x_1, x_2, \ldots, x_k.\varphi$ where φ has no quantifiers) is denoted as ∃PAD . The satisfiability of ∃PAD formulae was shown decidable in [12] and in NP [13]. Given a set $S \subseteq \mathbb{N}^k$ we say that S is ∃PAD ***definable*** if there is a finite set R of ∃PAD formulae, each formula with free variables $x_1, \ldots x_k$ such that $(n_1, \ldots, n_k) \in S \iff \bigvee_{\varphi \in R} \varphi(n_1, \ldots, n_k)$. Note that ∃PAD sets are closed under finite union, intersection and projection. It was shown in [4,5] that the reachability relation of parametric one-counter machines is ∃PAD definable.

Lemma 1 ([4], Lemma 4.2.2). *Given a parametric one-counter machine B and states s, t, the relation Reach$(B, s, t) = \{(x, y, n_1, \ldots, n_k) \mid (s, x) \to^* (t, y)$ in B^γ where $\gamma(p_i) = n_i\}$ is ∃PAD definable.*

2.2 Nonparametric Clock Elimination

Let A be a PTA. By modifying the region construction, we show how to build a PTA with equivalent halting problem without nonparametric clocks.

Once the value of a nonparametric clock c is above the largest constant appearing in A, the precise value of c does not affect any comparison. Now, the value of c is always a natural number. Hence, we eliminate nonparametric clocks by storing in the state space of C the values of the clocks up to the largest constant. However, we must ensure that the eliminated clocks progress simultaneously with the remaining parametric ones. This motivates 0/1 timed automata where the $+1$ updates correspond to the progress of time whereas the $+0$ updates correspond to taking an edge in A. Formally:

Lemma 2 ([1]). *Let $A = (S, s_0, C, P, F, E)$ be a PTA. Then there is a parametric 0/1 timed automaton $A' = (S', s'_0, C', P', F', E')$ such that $C' \subseteq C$ contains only parametrically constrained clocks of C and A and A' have equivalent halting problem. Moreover, $|A'| = O(2^{|A|})$.*

3 One Parametric Clock

For the rest of the section, fix a 1-PTA A. We show how to decide the halting problem for A. By Lemma 2, there is an exponentially larger parametric 0/1 automaton B with one (parametrically constrained) clock and equivalent halting problem. In Lemma 4 we show how to eliminate clock resets from B by introducing -1 edges, thereby turning B into a PBOCA. Hence, to decide the halting problem for A it suffices to decide the halting problem for a PBOCA with only $-1, 0, +1$ counter updates. We establish such a result in Theorem 5. Hence:

Theorem 3. *The halting problem for* 1*-PTAs is decidable in* NEXP*.*

Decidability of the halting problem 1-PTAs originally appeared in [1], albeit with nonelementary complexity. We give a completely different proof using one-counter machines yielding a NEXP algorithm. Later we show that the problem is also NEXP-hard. In the full version of the paper we prove the technical lemma:

Lemma 4. *Let B be a parametric 0/1 timed one-clock automaton. Then there is a PBOCA C such that B and C have equivalent halting problem. Further, all updates in C are either $-1, 0$ or $+1$ and $|C| = O(|B|)$.*

3.1 Decidability for Counter Machines with Constant Updates

We now show how to decide the halting problem for PBOCAs with all counter updates either $-1, 0$ or $+1$. Fix such a machine C. To show that C halts, we have to find an assignment γ and an accepting run π in C^γ. Even without knowing γ, we show that π splits into subruns of a simple form independent of γ the existence of which is reducible to satisfiability of certain ∃PAD formulae.

Let γ be a parameter assignment and assume that we guessed the order of parameters, let's say, $\gamma(p_1) < \gamma(p_2) < \ldots < \gamma(p_k)$, but not their precise values. Let $\boldsymbol{c_1}$ and $\boldsymbol{c_2}$ be arbitrary configurations of C^γ such that $\boldsymbol{c_1} \to^* \boldsymbol{c_2}$ in C^γ and consider a shortest run $\pi : \boldsymbol{c_1} \to \boldsymbol{c_2}$. There is a constant $M \in \mathbb{N}$, determined in Lemma 7, such that the run π can be factored into subruns between successive parameters and subruns around individual parameters. Formally, $\pi = \pi_0 \to \pi_1 \to \pi_2 \to \cdots \to \pi_l$ such that (π_0 can be possible empty)

- Even-indexed runs: $\gamma(p) - M \leq \mathit{counter}(\pi_{2i}) \leq \gamma(p) + M$ for a parameter p,
- Odd-indexed runs: $\gamma(p_r) + M < \mathit{counter}(\pi_{2i+1}) < \gamma(p_{r+1}) - M$ for some consecutive parameters $\gamma(p_r) < \gamma(p_{r+1})$,
- For every i, the runs π_i and π_{i+1} are joined by an edge $\mathit{end}(\pi_i) \to \mathit{start}(\pi_{i+1})$.

Notice that every edge in C changes the counter by at most 1. Hence, we have $\mathit{counter}(\mathit{start}(\pi_{2i+1})) = p_r + M + 1$ or $\mathit{counter}(\mathit{start}(\pi_{2i+1})) = p_{r+1} - M - 1$. Thus, $\mathit{start}(\pi_i)$ is always of the form $\mathit{start}(\pi_i) = (s_i, p_{f(i)} + x_i)$ for some state s_i, some $|x_i| \in \{M, M+1\}$ and parameter $p_{f(i)}$. Hence, $\mathit{start}(\pi_i)$ is uniquely determined by the triple $(s_i, f(i), x_i)$. Similarly, $\mathit{end}(\pi_i)$ is uniquely determined by some triple $(t_i, g(i), y_i)$ with $|y_i| \in \{M, M+1\}$.

By minimality, π visits every configuration only once. Hence an odd-indexed run can start in only one of $2nk$ configurations (n states, k parameters). Hence, the number of odd-indexed runs, and hence the total number of runs is $O(nk)$.

To show that there is a run from $\boldsymbol{c_1}$ to $\boldsymbol{c_2}$ we guess a factoring of the above form. We shall show (justifying the choice of M) in Lemma 8 that the odd-indexed runs π_{2i+1} correspond to runs in some one-counter machine $C_{h(2i+1)}$. By Lemma 1, the existence of a run in $C_{h(2i+1)}$ is ∃PAD expressible as: $\varphi_{2i+1} = \mathit{Reach}(C_{h(2i+1)}, s_{2i+1}, t_{2i+1})(n_{f(2i+1)} + x_{2i+1}, n_{g(2i+1)} + y_{2i+1}, n_1, \ldots, n_k)$.

In Lemma 9, we show that the even-indexed runs are independent of γ, can be precomputed and the reachability relation can be hardwired into the formula. Thus, we express the existence of a particular factoring from $\boldsymbol{c_1}$ to $\boldsymbol{c_2}$ as $\varphi =$

$\bigwedge_i \varphi_{2i+1} \wedge \psi(f, g, h, \overrightarrow{s}, \overrightarrow{t}, \overrightarrow{x}, \overrightarrow{y}) \wedge \bigwedge_i (n_i + M < n_{i+1})$ where the middle term encodes that the odd- and even-indexed runs are adjacent (directly computable) and that the even-indexed runs are valid (Lemma 9). The last conjunct encodes the technical restriction $\gamma(p_i) + M < \gamma(p_{i+1})$ imposed in Lemmas 8 and 9.

The restriction is relaxed as follows. First, if the parameters are not in the increasing order $\gamma(p_i) < \gamma(p_{i+1})$ then we relabel the parameters and build the appropriate formula. If $\gamma(p_i) \leq \gamma(p_{i+1}) < \gamma(p_i)+M$ then M depends only on $|C|$ (Lemma 7) and so only finitely many possibilities exist for $\gamma(p_{i+1})-\gamma(p_i)$. Hence we replace each occurrence of p_{i+1} in C by $p_i + w$ for the appropriate $w < M$.

Theorem 5. *Given states $s, t \in C$ the set $G(C, s, t) = \{(x, y, n_1, \ldots, n_k) \mid (s, x) \to^* (t, y)$ in C^γ where $\gamma(p_i) = n_i\}$ is* ∃PAD *definable.*

Recall that satisfiability of ∃PAD formulae is in NP [13] and that $|C|$ is exponential in $|A|$ (Lemmas 2 and 4). Hence, Theorem 3 follows. We have also proved the corresponding lower bound, in fact, already for a single parameter.

Theorem 6. *The halting problem for* 1*-PTAs with one parameter is* NEXP*-hard.*

The proof of ∃PAD definability relied on two lemmas that we prove now. First, we show how to calculate the odd-indexed runs. Let $\boldsymbol{c_1}, \boldsymbol{c_2}$ be configurations of C^γ between two successive parameters: $\gamma(p_i) < \mathit{counter}(\boldsymbol{c_1}), \mathit{counter}(\boldsymbol{c_2}) < \gamma(p_{i+1})$.

Consider the counter machine C_i obtained from C by evaluating all comparisons as if the counter was between $\gamma(p_i)$ and $\gamma(p_{i+1})$. Formally, C_i is obtained from C by removing all $\geq p_j$ and $\leq p_k$ edges for $k \leq i < j$ and all $\leq p_j$ and $\geq p_k$ edges for $k \leq i < j$ are replaced by $+0$ edges. Further, for $i > 0$ and $c \in \mathbb{N}$ we also remove all $\leq c$ edges from C_i. Note that the definition of C_i's depends only on the order of parameters in γ.

During a run $\pi : \boldsymbol{c_1} \to \boldsymbol{c_2}$ in C_i, the counter can become less than $\gamma(p_i)$ or greater than $\gamma(p_{i+1})$. So π does not necessarily correspond to a run in C. However, notice that C_i is a one-counter machine without parameters or $\leq x$ comparisons, i.e. an ordinary one-counter machine and thus has the following property [11]: If there is a run between two configurations then there is a run where the counter does not deviate much from the initial and the final counter value:

Lemma 7 ([11], Lemma 42). *Let C_i be as above. There is a constant M (polynomial in $|C_i|$) s.t. for any configurations $\boldsymbol{c_1}$ and $\boldsymbol{c_2}$ of C_i if $\boldsymbol{c_1} \to^* \boldsymbol{c_2}$ then there is a run $\pi : \boldsymbol{c_1} \to \boldsymbol{c_2}$ such that $U - M \leq \mathit{counter}(\pi) \leq V + M$ where $U = \min(\mathit{counter}(\boldsymbol{c_1}), \mathit{counter}(\boldsymbol{c_2}))$ and $V = \max(\mathit{counter}(\boldsymbol{c_1}), \mathit{counter}(\boldsymbol{c_2}))$.*

So as long as $\gamma(p_1) + M < \mathit{counter}(\boldsymbol{c_1}), \mathit{counter}(\boldsymbol{c_2}) < \gamma(p_2) - M$, the runs $\boldsymbol{c_1} \to \boldsymbol{c_2}$ in C_i correspond to runs in C. See the full version for the proof:

Lemma 8. *Let γ be an assignment with $\gamma(p_i) + M < \gamma(p_{i+1})$ for all i. Let $\boldsymbol{c}, \boldsymbol{d}$ be configurations with $\gamma(p_i) + M < \mathit{counter}(\boldsymbol{c}), \mathit{counter}(\boldsymbol{d}) < \gamma(p_{i+1}) - M$. Then $(\boldsymbol{c}, \boldsymbol{d}) \in C^\gamma(\gamma(p_i), \gamma(p_{i+1})) \iff \boldsymbol{c} \to^* \boldsymbol{d}$ in C_i^γ.*

For the even-indexed runs, the reachability around individual parameters, i.e. in intervals $(\gamma(p_i)-M, \gamma(p_i)+M)$, can be precomputed. Suppose that $\gamma(p_{i-1}) < \gamma(p_i)-M < \gamma(p_i)+M < \gamma(p_{i+1})$ so that the interval $(\gamma(p_i)-M, \gamma(p_i)+M)$ does not contain $\gamma(p_{i-1})$ or $\gamma(p_{i+1})$. Let $-M < x, y < M$ and let π be a run from $(s, \gamma(p_i)+x)$ to $(t, \gamma(p_i)+y)$ such that $\gamma(p_i) - M \leq \mathit{counter}(\pi) \leq \gamma(p_i) + M$. Then for every component $\pi(i)$, we can write $\mathit{counter}(\pi(j)) = \gamma(p_i) + z_j$ for some $-M \leq z_j \leq M$. But now, the run π is valid for any specific value of $\gamma(p_i)$ as only z_j determines which edges are enabled in C^γ. (See the full version)

Lemma 9. *Let γ, δ be parameter assignments with $\gamma(p_i)+M < \gamma(p_{i+1}), \delta(p_i)+M < \delta(p_{i+1})$ for all i. Let $s, t \in c$ be states and $-M < x, y < M$ integers. Then*

$$((s, \gamma(p_i)+x), (t, \gamma(p_i)+y)) \in C^\gamma(\gamma(p_i) - M, \gamma(p_i)+M) \iff$$
$$((s, \delta(p_i)+x), (t, \delta(p_i)+y)) \in C^\delta(\delta(p_i) - M, \delta(p_i)+M)$$

Furthermore, it is decidable in polynomial time whether $((s, \gamma(p_i)+x), (t, \gamma(p_i)+y)) \in C^\gamma(\gamma(p_i) - M, \gamma(p_i)+M)$ for any (and all) such assignment γ.

4 Two Parametric Clocks

We now show that the halting problem for 2-PTAs is equivalent to the halting problem for PBOCAs. The equivalence is used in Section 4.2 to show decidability of the halting problem in certain classes of 2-PTAs.

First, observe that a counter can be stored as a difference of two clocks, which can be used (see the full version) to show the easier direction of the equivalence.

Theorem 10. *Let C be a PBOCA. Then there is a 2-PTA A such that A and C have equivalent halting problem. Moreover, if C has no '$\equiv 0 \bmod c$' edges then A has no nonparametric clocks. Otherwise, A has one nonparametric clock.*

4.1 Reduction to Parametric Bounded One-Counter Machines

For the converse, fix A to be a 2-PTA. We reduce A to a PBOCA C. To begin, we construct (Lemma 2) a parametric 0/1 timed automaton B with two parametrically constrained clocks, denoted x and y, with the halting problem equivalent to A. We then transform B to C. Denote the counter of C by z.

For the time being, we need to relax the assumption that z stays nonnegative. That is, subtracting 5 when the counter is 2 results in the counter being -3. In Remark 12 we later show how to restore the nonnegativity of the counter.

The idea of the reduction is that, after a clock of B is reset, that clock equals zero, so we use z to store the value of the other clock. We construct C in such a way that after a reset of y, counter z stores the value of x and after a reset of x, counter z stores $-y$. Initially C starts with the counter equal to 0.

Machine C then operates in phases. Each phase corresponds to a run of B between two consecutive resets of some (possibly different) clock.

Suppose y was the last clock to reset. After the reset, the configuration of B is $(s, (z, 0))$ for some state $s \in B$ and the counter $z = x$. We show how C calculates the configuration after the next clock reset in B.

After time Δ, the clocks go from configuration $(z, 0)$ to $(z + \Delta, \Delta)$. Based on the guards, different edges in B^γ are enabled as time progresses. Precisely, suppose we know the order of the parameters $p_1 < p_2 < \ldots < p_k$. Then let ***region*** $R_{(i,j)}$ be the set of clock valuations $[p_i, p_{i+1}] \times [p_j, p_{j+1}]$. Then the set of enabled edges depends only on the region $R_{(i,j)}$ the clocks (x, y) lie in.[2]

Therefore, machine C guesses the regions $R_{(i_0,j_0)}, R_{(i_1,j_1)}, \ldots, R_{(i_m,j_m)}$ in the order in which they are visited by the clocks (x, y) and it also guesses the states $s_0, s_1, \ldots, s_m$ of B when each region R_l is visited for the first time, the state t in which the next reset occurs and which clock is reset next (see Fig 2).

Machine C checks that the sequence is valid as follows. First, C checks, that $(z, 0)$ lies in R_0. Second, it checks that the regions are adjacent: $i_{l+1} - i_l = 1 \wedge j_{l+1} = j_l$ or $i_{l+1} = i_l \wedge j_{l+1} - j_l = 1$ or $i_{l+1} - i_l = j_{l+1} - j_l = 1$. The last case corresponds to the clocks hitting a corner of a region. Then, C checks that starting in clock configuration $(z, 0)$, the regions can be visited in the guessed order.

Consider region $R_{(u,v)}$ for some u, v. When the region is visited for the first time, then either clock x equals p_u or clock y equals p_v. In the former case, the clock configuration is $(p_u, p_u - z)$, in the latter case, it is $(p_v + z, p_v)$. The configuration depends on the direction in which $R_{(u,v)}$ is visited. See Fig. 2.

- If $i_{l+1} - i_l = 1$ then C checks that clock x reaches $p_{i_{l+1}}$ before clock y reaches p_{j_l+1}. That is, $p_{i_{l+1}} - z \leq p_{j_l+1}$. Equivalently, $p_{i_{l+1}} \leq z + p_{j_l+1}$, which can be easily tested by a PBOCA. In Fig. 2 this corresponds to region $R_{(1,0)}$, which is visited before $R_{(2,0)}$.
- Similarly, if $j_{l+1} - j_l = 1$. E.g, in Fig. 2 region $R_{(2,1)}$ is visited before $R_{(2,2)}$.

We say $R_{(u,v)}$ was ***reached from left*** in the first and that $R_{(u,v)}$ was ***reached from bottom*** in the second case. See Fig. 2 for the intuition behind the names.

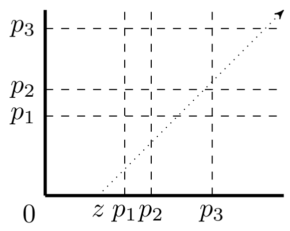

Fig. 2. Regions for parameters $p_1 < p_2 < p_3$. The dotted line shows an evolution of clock configuration, which visits $R_{(0,0)}, R_{(1,0)}, R_{(2,0)}, R_{(2,1)}, R_{(2,2)}, R_{(3,2)}, R_{(3,3)}$.

Finally, C checks reachability within individual regions. For $l = (u, v)$, let $\boldsymbol{c_l}$ be the configuration in which the region R_l is visited for the first time. Then C checks that a run from $\boldsymbol{c_l}$ to $\boldsymbol{c_{l+1}}$ exists in R_l.

Now, with each $R_{(i,j)}$, we introduce a one-counter machine $B_{(i,j)}$ obtained from B assuming clock $x \in [p_i, p_{i+1}]$ and clock $y \in [p_j, p_{j+1}]$, instantiating all comparisons accordingly and by removing all edges resetting a clock. Each $B_{(i,j)}$ corresponds to the region $R_{(i,j)}$ in the same way automata C_i corresponded to one-dimensional regions in Section 3.

[2] Our definition of rectangular regions differs slightly from the one usually given in the literature. However, as all inequalities are nonstrict the regions are sufficient. For ease of presentation, we also use the convention $p_0 = 0$ and $p_{k+1} = \infty$.

Notice that $B_{(i,j)}$'s are 0/1 automata without resets or comparisons, i.e. one-counter machines. In particular, the reachability relation for $B_{(i,j)}$'s is semilinear. For a pair of states s and t of a one-counter machine X define $\Pi(X,s,t)$ to be the set of counter values reachable at t by a run starting in state s and counter equal to 0: $\Pi(X,s,t) = \{v \mid \exists \pi \in X. \mathit{start}(\pi) = (s,0) \wedge \mathit{end}(\pi) = (t,v)\}$.

Lemma 11. *Let X be a one-counter machine with 0/1 updates. Then for any states $s,t \in X$ the set $\Pi(X,s,t)$ is effectively semilinear: $\Pi(X,s,t) = N \cup \bigcup_{j=1}^{j=r}\{a_j + b_j\mathbb{N}\}$ where $N \subseteq \mathbb{N}$ is finite and $a_j, b_j \in \mathbb{N}$.*

Now, to check that a run from $\boldsymbol{c_l}$ to $\boldsymbol{c_{l+1}}$ exists in R_l, machine C distinguishes whether R_l and R_{l+1} are reached from bottom or from left and uses the semilinearity of the reachability relation of the corresponding $B_{(i,j)}$.

The translation is mundane and is given in the full version of the paper. For example, suppose $R_l = R_{(p_x,p_y)}$ for parameters p_x and p_y. Then $\boldsymbol{c_l} = (s_l,(p_x,p_x - z))$ or $\boldsymbol{c_l} = (s_l,(p_y + z, p_y))$ depending on the direction. If R_l was reached from left and R_{l+1} from bottom then C checks that $(s_{l+1},(p_{y+1} + z, p_{y+1}))$ is reachable from $(s_l,(p_x,p_x - z))$. That is, that $z + p_{y+1} - p_x \in \Pi(B_l, s_l, s_{l+1})$. All such constraints can be checked using '$\equiv 0 \bmod c$' edges (see Fig. 3).

Finally note that once the value of a clock becomes larger than p_k its exact value is irrelevant to any future comparison. Hence, C tracks x and y only up to p_k and remembers which clocks exceed it. Hence, we can assume that the counter of C is always inside $[-p_k, p_k]$.

$$\to \xrightarrow{+p_x} \xrightarrow{-p_y} \xrightarrow{-a} \xrightarrow{\equiv 0 \bmod b} \xrightarrow{+a} \xrightarrow{+p_y} \xrightarrow{-p_x}$$

Fig. 3. Gadget testing that for given $a,b \in \mathbb{N}$ there is $k \in \mathbb{N}$ such that $z + p_x - p_y = a + kb$,i.e. $z + p_x - p_y - a \equiv 0 \bmod b$. Letter z denotes the current counter value.

Next, we modify C to ensure that the counter is always nonnegative. Let C' be obtained from C by adding a new initial state and a $+p_k$ edge from the new to the original initial state. Further, any comparison edge (s,G,t) (e.g., where G is $\leq p_i$) is replaced by a gadget of three edges $(s,-p_k,q)$, (q,G,q') and $(q',+p_k,t)$ which subtract p_k from the counter, perform the original check and then add p_k to the counter thereby offsetting the counter by p_k.

Remark 12. We can assume that the counter of C is always inside $[0, 2p_k]$.

Note that the construction depends on the order of parameters. However, we can build an automaton for each possible order, check the order of parameters and then transition into the automaton for the appropriate order.

Theorem 13. *Given a 2-PTA there is a PBOCA with equivalent halting problem.*

The reduction was inspired by [6] (see Related Work). Unlike [6], we exploit semilinearity in individual regions and perform one phase in a single stage of C.

4.2 The One-Parameter Case

Suppose that the 2-PTA A uses only a single parameter p and consider the corresponding PBOCA C. We show that all '$\equiv 0 \bmod c$' and '$+[0,p]$' edges can be eliminated from C. Using Remark 12, we show in Lemma 15 how to decide the halting problem in the resulting class of PBOCAs.

Inspecting the detailed proof of the reduction from A to C (as found in the full version), observe that '$+[0,p]$' edges are introduced only when two successive regions are both visited from left or both visited from bottom. For a single parameter, only regions $[0,p] \times [0,p], [0,p] \times [p,\infty], [p,\infty] \times [0,p], [p,\infty] \times [p,\infty]$ exist. Simple case analysis shows that this can occur only when the counter starts at 0—this can be treated separately thereby eliminating '$+[0,p]$' edges from C.

Next, we also eliminate '$\equiv 0 \bmod c$' edges from C. Intuitively, C shall store in its state space the counter modulo c_i for each $c_1, \ldots, c_r$ appearing as '$\equiv 0 \bmod c_i$' in C. The construction depends on the value of $p \bmod c_i$ for each i.

Given $D = (d_1, \ldots, d_r)$, let C_D be the one-counter machine obtained from C which tracks the counter modulo each c_i assuming $p \equiv d_i \bmod c_i$. Formally, the states of C_D are $S \times \mathbb{Z}_{c_1} \times \ldots \times \mathbb{Z}_{c_r}$ where S are the states of C and $\mathbb{Z}_{c_i}$ denotes the ring of integers modulo c_i. The machine C_D contains all comparison edges of C. Further, let $(v_1, \ldots, v_r) \in \mathbb{Z}_{c_1} \times \ldots \times \mathbb{Z}_{c_r}$. Let E be the edges of C, then C_D also contains the following edges:

- $((q, v_1, \ldots, v_r), \pm c, (q', v_1 \pm c, \ldots, v_r \pm c)$ if $(q, \pm c, q') \in E$,
- $((q, v_1, \ldots, v_r), \pm p, (q', v_1 \pm d_1, \ldots, v_r \pm d_r)$ if $(q, \pm p, q') \in E$,
- $((q, v_1, \ldots, v_r), +0, (q', v_1, \ldots, v_r))$ if $v_i = 0$ and $(q, \equiv 0 \bmod c_i, q') \in E$.

Notice that there are no '$\equiv 0 \bmod c$' edges in C_D. By construction, runs in C_D^γ are equivalent to runs C^γ provided $d_i \equiv \gamma(p) \bmod c_i$. That is:

Lemma 14. *Let γ be an assignment such that $\gamma(p) = d_i \bmod c_i$ for each i. Let $(s,x), (t,y)$ be configurations of C. Then $(s,x) \to^* (t,y)$ in C^γ if and only if $((s, x \bmod c_1, \ldots, x \bmod c_r), x) \to^* ((t, y \bmod c_1, \ldots, y \bmod c_r), y)$ in C_D^γ.*

For given D, finding an accepting run π such that $counter(\pi) \leq 2 \cdot \gamma(p)$ suffices (Remark 12) to decide the halting problem for C_D. For any such run π and index i we can write $counter(\pi(i)) = a\gamma(p) + b$ where $a \leq 2$ and $b < \gamma(p)$.

Since a is bounded, we can build a one-counter machine G keeping a in the state space and b in the counter. We do not enforce $b < \gamma(p)$ (or any other $\leq x$ constraint) in G. Instead, we use Lemma 7 on G and split π into subruns close to and far from a multiple of $\gamma(p)$. We write $\pi = \tau_0 \to \pi_1 \to \tau_1 \ldots \pi_l \to \tau_l$ such that for every τ_i the value $counter(\tau_i) \bmod \gamma(p) \in [0, \ldots, M] \cup [\gamma(p) - M, \gamma(p))$. For every π_i we have $counter(\pi_i) \bmod \gamma(p) \in (M, \gamma(p) - M)$. Then we use techniques on factoring of runs analogous to those used for one 1-PTAs (Section 3.1). In general, we have: (See the full version)

Lemma 15. *Given C with one parameter p, no '$\equiv 0 \bmod c$' and no '$+[0,p]$' edges, $k \in \mathbb{N}$ and states $s, t \in C$ the set $G(C,s,t,k) = \{(x,y,q) \mid \exists \pi : (s,x) \to (t,y) \in C^\gamma$ s.t. $counter(\pi) \leq k \cdot q$ where $q = \gamma(p)\}$ is $\exists$PAD definable.*

Theorem 16. *The halting problem is decidable for 2-PTAs with one parameter.*

This settles the case of 2-PTAs with a single parameter. However, even the case of only two parameters is open. On the other hand, already for a single parameter, we have the following lower bound. (See the full version)

Theorem 17. *The decidability of the halting problem for 2-PTAs with a single parameter is* $\mathsf{PSPACE}^{\mathsf{NEXP}}$*-hard.*

Acknowledgments. This research was financially supported by EPSRC.

References

1. Alur, R., Henzinger, T.A., Vardi, M.Y.: Parametric real-time reasoning. In: Proceedings of the 25th Annual Symposium on Theory of Computing. ACM Press (1993)
2. Doyen, L.: Robust parametric reachability for timed automata. Information Processing Letters 102(5), 208–213 (2007)
3. Fearnley, J., Jurdziński, M.: Reachability in two-clock timed automata is PSPACE-Complete. In: Fomin, F.V., Freivalds, R., Kwiatkowska, M., Peleg, D. (eds.) ICALP 2013, Part II. LNCS, vol. 7966, pp. 212–223. Springer, Heidelberg (2013)
4. Haase, C.: On the Complexity of Model Checking Counter Automata. PhD thesis, University of Oxford (2012)
5. Haase, C., Kreutzer, S., Ouaknine, J., Worrell, J.: Reachability in succinct and parametric one-counter automata. In: Bravetti, M., Zavattaro, G. (eds.) CONCUR 2009. LNCS, vol. 5710, pp. 369–383. Springer, Heidelberg (2009)
6. Haase, C., Ouaknine, J., Worrell, J.: On the relationship between reachability problems in timed and counter automata. In: Finkel, A., Leroux, J., Potapov, I. (eds.) RP 2012. LNCS, vol. 7550, pp. 54–65. Springer, Heidelberg (2012)
7. Henzinger, T.A., Manna, Z., Pnueli, A.: What good are digital clocks? In: Kuich, W. (ed.) ICALP 1992. LNCS, vol. 623, pp. 545–558. Springer, Heidelberg (1992)
8. Hune, T., Romijn, J., Stoelinga, M., Vaandrager, F.: Linear parametric model checking of timed automata. In: Margaria, T., Yi, W. (eds.) TACAS 2001. LNCS, vol. 2031, pp. 189–203. Springer, Heidelberg (2001)
9. Ibarra, O.H., Jiang, T., Trân, N., Wang, H.: New decidability results concerning two-way counter machines and applications. In: Lingas, A., Carlsson, S., Karlsson, R. (eds.) ICALP 1993. LNCS, vol. 700, pp. 313–324. Springer, Heidelberg (1993)
10. Jovanović, A., Lime, D., Roux, O.H.: Integer parameter synthesis for timed automata. In: Piterman, N., Smolka, S.A. (eds.) TACAS 2013. LNCS, vol. 7795, pp. 401–415. Springer, Heidelberg (2013)
11. Lafourcade, P., Lugiez, D., Treinen, R.: Intruder deduction for AC-like equational theories with homomorphisms. In: Research Report LSV-04-16, LSV, ENS de Cachan (2004)
12. Lipshitz, L.: The Diophantine Problem for Addition and Divisibility. Transactions of the American Mathematical Society, 235 (1978)
13. Lipshitz, L.: Some remarks on the diophantine problem for addition and divisibility, vol. 33 (1981)
14. Miller, J.S.: Decidability and complexity results for timed automata and semi-linear hybrid automata. In: Lynch, N., Krogh, B.H. (eds.) HSCC 2000. LNCS, vol. 1790, pp. 296–310. Springer, Heidelberg (2000)
15. Ouaknine, J., Worrell, J.B.: Universality and language inclusion for open and closed timed automata. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 375–388. Springer, Heidelberg (2003)